BAREMETAL APPLICATION ON ARM VERSTILEPB REPORT

--First of all, this BareMetal SW without os and running this SW without IDE.

- SO, we need to write and execute every driver

objectives

Create a SW to send a “learn-in-depth<Ahmed>” using uart

tools

-notepad++ or sublime

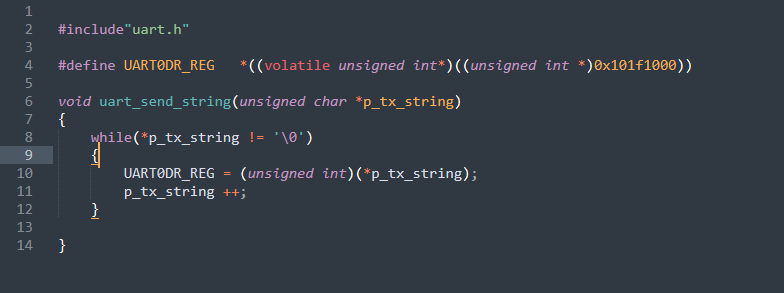
--cross toolchain to compile like arm-non-eapi toolchain

-- emulator to have versatile board like qemu

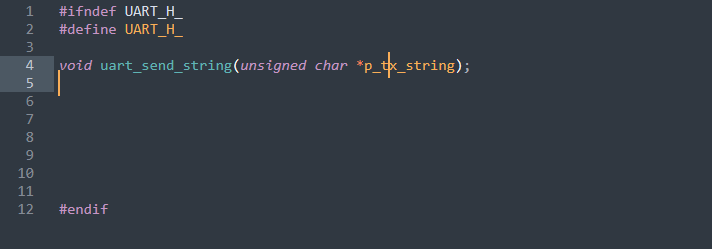
Source code

* App.c
* Uart.h
* Uart.c
* Startup.s
* Linker\_ script. ld.

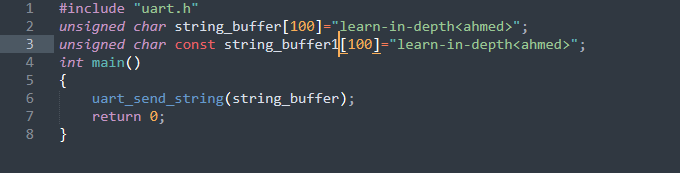
Uart.c



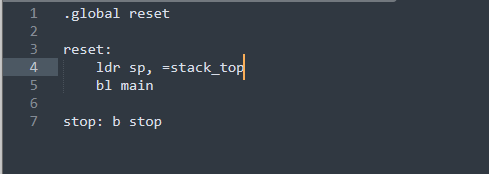
Uart.h



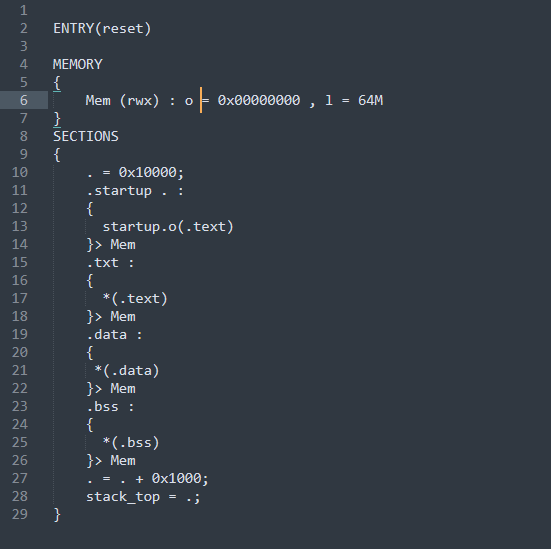
App.c



Startup.c



Linker\_script.ld



Compilation process

Pre-processing and compiling

$ arm-none-eabi-gcc.exe -c -I -g -mcpu=arm926ej-s app.c -o app.o

$ arm-none-eabi-gcc.exe -c -I -g -mcpu=arm926ej-s uart.c -o uart.o

Assembling

$ arm-none-eabi-as.exe -mcpu=arm926ej-s startup.s -o startup.o

Linking and map\_file

$ arm-none-eabi-ld.exe -T linker\_script.ld startup.o app.o uart.o -o learn-in-depth.elf -Map=map\_file.map

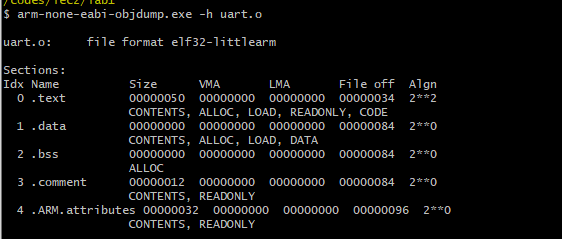
Binary\_file:

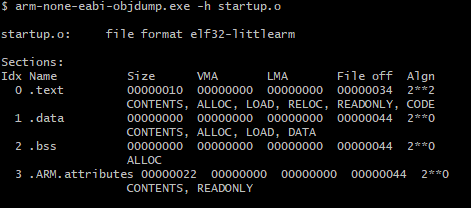
$ arm-none-eabi-objcopy.exe -O binary learn-in-depth.elf learn-in-depth.bin

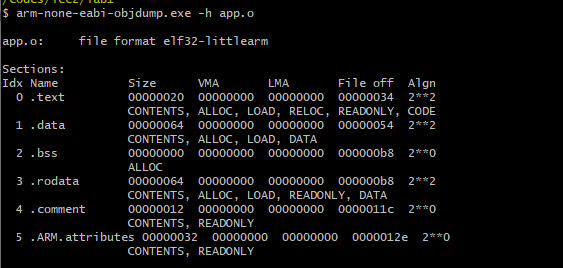
Run in qemu simulator

$ qemu-system-arm -M versatilepb -m 128M -nographic -kernel learn-in-depth.bin

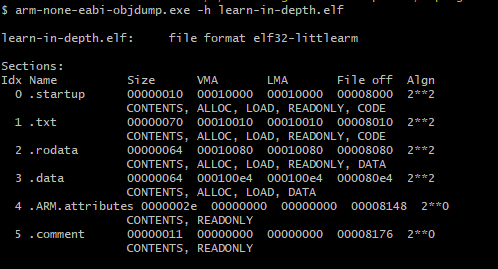
Sections of relocatable locations



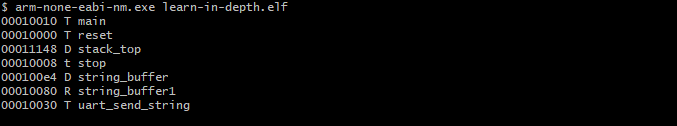




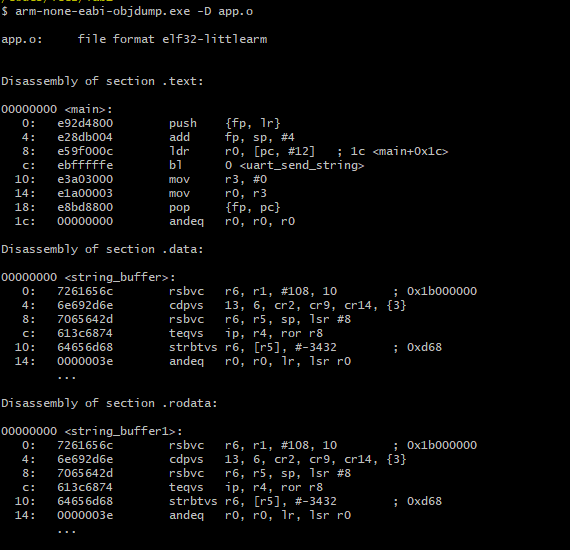
Run\_time\_addresses



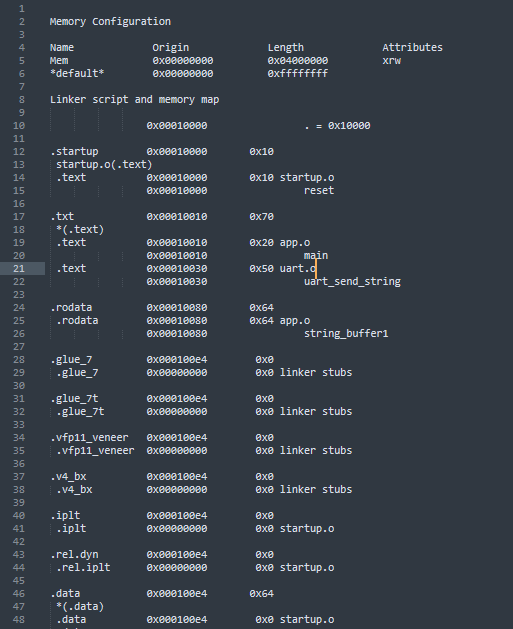
Symbols of elf file



Disassembly



Map\_file



Qemu output

